10. ročník
Žilina 2. – 4. 7. 2018
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2024: 1–6

PF2021! aneb Není šum jako šum

PAVEL STŘÍŽ (CZ)

Abstrakt. Článek stručně zmiňuje historii používání pseudonáhodných čísel ve 2D a 3D grafice.

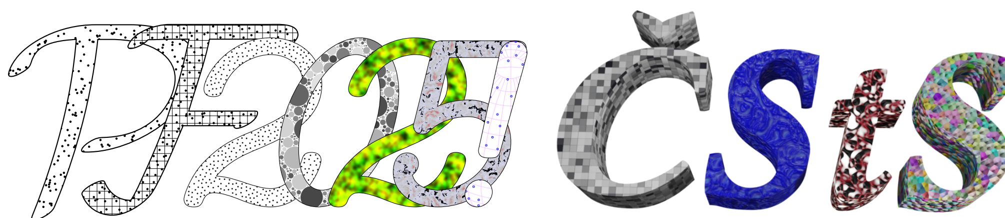
Klíčová slova. PRNG, 2D, 3D, grafika.

PF2021! OR DIFFERENT TYPES OF NOISE

Abstract. The article briefly mentiones the history of using pseudo-random numbers in 2D and 3D graphics.

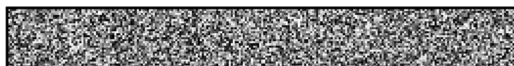
Keywords. PRNG, 2D, 3D, graphics.

Motto: [...] 12585 33893 43498 [...], str. 165, řádek 08210, sloupce 5–7.
A Million Random Digits with 100,000 Normal Deviates
`www.rand.org/pubs/monograph_reports/MR1418.html`



1. Úvodem pár vzpomínek

Mé první vzpomínky na práci s náhodností spadají pod Sinclair BASIC na základní škole, kdy jsme např. volili den v týdnu jako jedno z čísel 1 až 7 a dál s tím pracovali. Pár let poté jsem podobné příklady procvičoval v QBASICu s otevřenou knihou *Sbírka úloh z programování* od manželů Töpferových. V průběhu času se člověk dostal k fyzice a šumu u televizí až ke kosmickému záření.



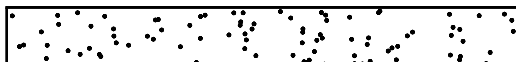
Zaujala mě práce Norberta Wienera, nejen jeho aplikace zpětné vazby, ale i práce nad Brownovým pohybem. To jsou ty známé základoškolské fyzikální pokusy vhození hypermanganu do vody. Dnes je bílý šum (angl. white noise) brán vážně ve všech významnějších oblastech přírodovědeckého bádání, včetně ekonomie, informatiky, statistiky a geometrie.

Z poslední doby se mi do ruky dostala kniha Davida Johnstona *Random Number Generators—Principles and Practices* z roku 2018, kde se to hemží kódy v C a Pythonu. Dle MSC2020 bychom naši článkovou rešerši začali asi v oblastech 60H40 (White noise theory), 65Cxx (Probabilistic methods…) či 11K45 (Pseudorandom numbers; Monte Carlo methods).

2. PF aneb Základ černobíle

Na první ukázce, písmenku $\mathcal{P}$, vidíme typickou situaci užití pseudonáhodných čísel ve 2D (angl. pseudo-random numbers). Bez ohledu na kvalitu statistických vlastností to není po vizuální stránce příjemné. Jsou tam prázdná oka, kruhy se mohou překrývat. K testování lze doporučit `www.random.org`. Zde je ukázka přes Ti**k**Z.

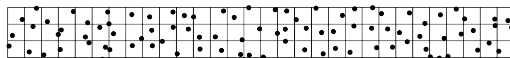
```
\documentclass{standalone}
\usepackage{tikz}
\tikzset{inner sep=0pt, outer sep=0pt}
\begin{document}
\begin{tikzpicture}
\foreach \x in {1,...,900} {
  \pgfmathparse{rnd*100} \let\malx=\pgfmathresult
  \pgfmathparse{rnd*100} \let\maly=\pgfmathresult
  \node[xshift=\malx mm, yshift=\maly mm, circle, fill, minimum width=1mm]{};
} % end of \foreach \x
\end{tikzpicture}
\end{document}
```



Druhou stranou mince by byla dokonalá mřížka z bodů. Spojení obou nápadů vzniká stratifikovaný výběr (angl. supersampling či jittered grid). Prvně si plochu rozdělíme na menší čtverce a v každém volíme po jednom bodu. Chceme-li bodů víc, zjemníme mřížku. Dostáváme písmenko $\mathcal{F}$. Vizuálně je to lepší, ale stále tam jsou místy body u sebe a občas řeky. To vadí především typografům z čteného textu odstavců. U mřížky se mohou objekty překrývat. Opět vzorek přes Ti**k**Z.

```
\documentclass{standalone}
\usepackage{tikz} \tikzset{inner sep=0pt, outer sep=0pt}
\begin{document}
\begin{tikzpicture}
\draw[step=3.3333mm] (0,0) grid (100mm,100mm);
\foreach \x in {1,...,30} {
  \foreach \y in {1,...,30} {
    \pgfmathparse{3.3333*(rnd-1+\x)} \let\malx=\pgfmathresult
    \pgfmathparse{3.3333*(rnd-1+\y)} \let\maly=\pgfmathresult
    \node[xshift=\malx mm, yshift=\maly mm, circle, fill, minimum width=1mm]{};
```

```
  } % end of \foreach \y
} % end of \foreach \x
\end{tikzpicture}
\end{document}
```

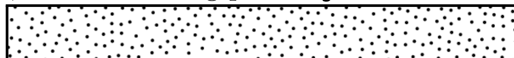


3. 2021 aneb Přes stupně šedi do barvy

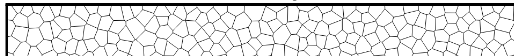
Grafici šli dál.

3.1. Robert Bridson

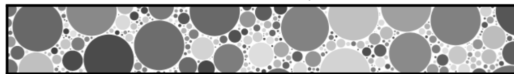
Jason Davies (zde, je znám hlavně díky mrakům slov) či Mike Bostock (zde, zakladatel serverů `bl.ocks.org` a `observablehq.com`) představují Bridsonův algoritmus (2007) generování bodů s geometrickým vztahem, že žádný nový bod nesmí být blíž než určená vzdálenost (angl. Poisson-disc sampling). Nelze již mluvit o pokusu náhodného generování, neb existuje mezi body vztah. Vizuálně je to pro lidské oko příjemné. Ukázka je v první číslici *2*. Davies vykresluje body přes canvas HTML5, Bostock do svg přes `D3js`.



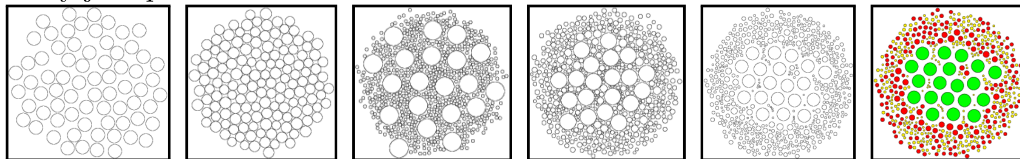
Ze středů lze snadno získat Voronoi diagram.



Zobecnění přináší algoritmus Mitchell's best-candidate, kdy se generuje sada k bodů a vybírá se z nich jen jeden, takový, který je nevzdálenější vůči všem ostatním. To dává možnost např. volit různý poloměr kruhů (číslice *0*).



Zájemci mohou nahlédnout na mé starší pokusy přes Lua, byť tedy užité algoritmy jsou pomalé.



3.2. Ken Perlin

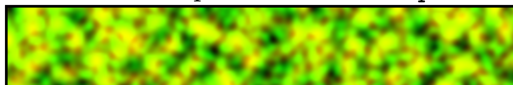
Mezník představuje Perlinův šum (angl. Perlin noise) z roku 1983 (příspěvek Kena Perlina z roku 1985 se jmenuje **An Image Synthesizer**), kdy dochází k interpolaci mezi body. Tím lze snadno vytvářet šum ve 2D a 3D, se zahrnutím

času či barvy i ve vyšších rozměrech. Zde je typická ukázka, vypadá to jak výškový model (angl. DEM) známý z geografických inf. systémů.



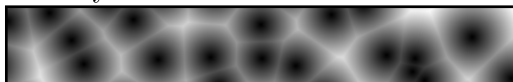
Za pozornost stojí i článek Ken Perlin a Fabrice Neyret: **Flow Noise**. Podobné výsledky dostáváme pomocí algoritmu Simplex noise a volně dostupné varianty OpenSimplex noise. Zde jsou dostupné implementace v Javě, JavaScriptu a nezapomínejme na Rust. Zaujal mě článek *Recursive Wang Tiles for Real-Time Blue Noise.* Pro studenty lze doporučit na YouTube kanál The Coding Train Daniela Shiffmana v jeho oblíbeném nástroji `p5js` a jeho knihu **The Nature of Code.**

Druhá cifra *2* vznikla v JavaScriptu v balíčku `simplex-noise` přes `npm`.

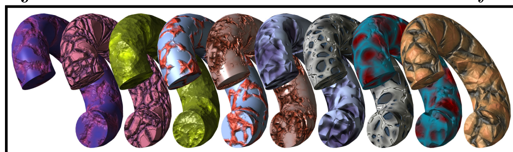


3.3. Steven Worley

Další skok přichází v roce 1996, kdy Steven Worley na konferenci představuje tvorbu procedurální textury.



Zde jsou ukázky z jeho článku *A cellular texture basis function.*

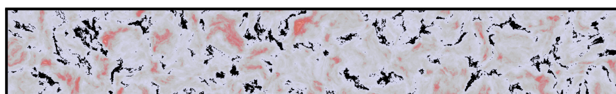


Poslední cifra, *1*, je z `examples/texturegranite.rs` z knihovny `noise` v0.6.0 na `crates.io`, přesně soubor `texture_granite_planar.png.` Získáváme malbu skoro jako od Jacksona Pollocka.

Po instalaci:

```
$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
$ echo "export PATH=$HOME/.cargo/bin:$PATH" >>~/.bashrc
$ source $HOME/.cargo/env
$ git clone https://github.com/Razaekel/noise-rs.git
$ cd noise-rs
$ cargo build
```

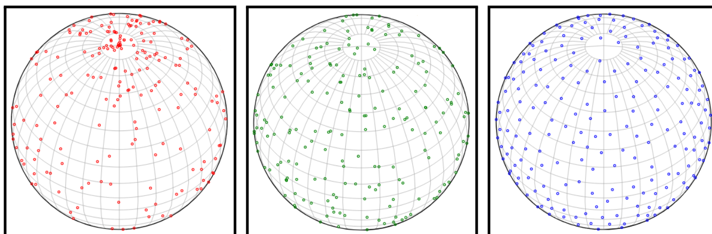
jsem spouštěl `cargo run --example texturegranite`.



Vážnějším zájemcům doporučuji knihu Patricio Gonzales Vivo a Jen Lowe: **The Book of Shaders** se zvýrazněnými ukázkami. Kniha vzniká od roku 2015 a je stále v přípravě a ve vývoji v OpenGL ES. Vážným zájemcům pak doporučuji knihy Ebert, Musgrave, Peachey, Perlin a Worley: *Texturing and Modeling: A Procedural Approach,* 3. vyd. z roku 2002 a z roku 2017 Tanya X. Short a Tarn Adams: *Procedural Generation in Game Design.*

4. Koule na místo vykřičníku

Rust má v ukázkách příklad vzniku textury pasovanou na kouli. Je to předchozí kód, další vygenerovaný soubor `texture_granite_sphere.png` ve složce `example_images.` Rovnovnoměrné rozdělení bodů na kouli je známý a vyřešený problém, viz MathWorld. Jason Davies srovnává intuitivní řešení ve sférické soustavě souřadnic (vlevo), řešení s korekcí (uprostřed) a aplikovaný Mitchellův algoritmus výběru nejlepšího kandidáta (koule vpravo). Charakteristikami se dostáváme na hranici modrého šumu.



5. ČStS aneb Vzorky pomocí procedurálních textur

Situace se komplikuje, pokud zvolíme obecný 3D objekt.

Existuje řada programů, které umí pracovat s texturami. Mezi nejvýraznější svobodné programy patří Blender (zkr. BS). Ten je mezi námi už od roku 1988, jen o 10 let mladší než TEX (1978) a o 5 let starší než R (1993). Některé postupy jsou vlastní, některé inspirované jinými programy na 3D grafiku. Jednou z inspirací byl program Filter Forge, kde se užívá jazyk Lua.

Za pozornost dávám knihu Richarda Egdahla **Texture Magic: Procedural Textures for Blender Cycles,** kterou považuji za představitele Blenderu do verze 2.79b. Důležité je, že Blender je specialista na 3D a položení textur na libovolné 3D objekty je přirozený krok grafiků.

Blender má nadstavbu Sverchok (zkr. SV, rusky сверчок) na parametricky definované 3D objekty. Příchod nadstavby Animation Nodes (zkr. AN) znamená mezník u animování s následnou možností úpravy textů.

Blender udělal obří skok od verze 2.80 s tahem k nástroji Everything Nodes, kdy přes grafické rozhraní (angl. shader nodes) by měly být dostupné téměř všechny nástroje Blenderu, speciálně systém částic.

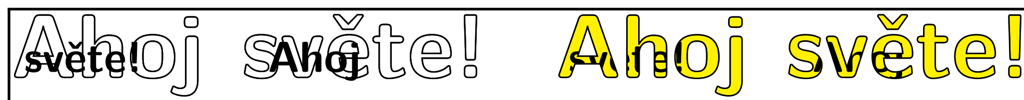
Lze doporučit YouTube kanál LiveNoding od Jimma Gunawaneho.

Znaky v ČStS jsou vytvořené v Blenderu za pomocí procedurálních textur a vyrenderovány jako rastrové obrázky: Č přes White Noise Texture, S přes Noise Texture, t přes Musgrave Texture a S pomocí Voronoi Texture.

6. Kompletace novoročenky místo Závěru

Pro TEXisty bude zajímavá první část. Přes TikZ vkládám pozadí dovnitř znaků. Zde je minimální ukázka „Ahoj světe!“ bez a s vyříznutím. Dávám na sebe pozadí velkých slov, malá slova a obrys velkých znaků.

```
\documentclass{article}
\pagestyle{empty}
\usepackage{tikz}
\begin{document}
\def\ukaz#1#2#3#4{%
\begin{tikzpicture}[text height=2ex, text depth=1ex]
\node{\Huge\bfseries\sffamily
  \pgfsetstrokecolor{black}\pgfsetfillcolor{yellow}%
  \pdfextension literal {#1 Tr}%
  \makebox[0pt][l]{#2}%
  \makebox[0pt][l]{\large
    \pdfextension literal {#3 Tr}%
    \tikz[trim left, baseline=0mm]
      \tikz{\node[xshift=7mm, yshift=1.5mm, text height=2ex, text
    depth=1ex]{\color{black}#4};
      };% end of \tikz
  }% end of \makebox
  \pdfextension literal {1 Tr 0.4 w}%
  \makebox[0pt][l]{#2}%
  };% end of \node
\end{tikzpicture}%
}% end of \ukaz
\ukaz{1}{Ahoj}{0}{světe!}\kern2cm \ukaz{1}{světe!}{0}{Ahoj}\kern28mm
\ukaz{6}{Ahoj}{0}{světe!}\kern2cm \ukaz{6}{světe!}{0}{Ahoj}
\end{document}
```



[…] but with Perlin noise I may pick numbers like this: 2, 3, 4, 3, 4, 5, 6, 5, 4, 5, 6, 7 […]

Daniel Shiffman @ Perlin Noise and Flow Fields
`www.youtube.com/watch?v=sor1nwNIP9A`

Kontaktní adresa

Ing. Pavel Stříž, Ph.D., U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: `pavel@striz.cz`